

Encoding Temporal Markov Dynamics in Graph for Visualizing and Mining Time Series

Lu Liu

University of Maryland, Baltimore County
Baltimore, Maryland
Email: liuluny@umbc.edu

Zhiguang Wang

GE Global Research
San Ramon, California
Email: zgwang813@gmail.com

Abstract—Time series is attracting more attention across statistics, machine learning and pattern recognition as it appears widely in both industry and academia, but few advances have been achieved in effective time series visualization due to its temporal dimensionality and complex dynamics. Inspired by recent effort on using network metrics to characterize time series for classification, we present an approach to visualize time series as complex networks based on the first order Markov process in its temporal ordering. In contrast to the classical bar charts, line plots and other statistics based graph, our approach delivers more intuitive visualization that better preserves both the temporal dependency and frequency structures. It provides a natural inverse operation to map the graph back to time series, making it possible to use graph statistics to characterize time series for better visual exploration and statistical analysis. Our experimental results suggest the effectiveness on various tasks such as system identification, pattern mining and classification on both synthetic and the real time series data.

I. INTRODUCTION

Time series data is ubiquitous in both industry and academia. Except for the perceptual signals that we are able to easily understand such as audio, video and nature language, most of the other time series generated are mechanical signals. Therefore, Understanding temporal patterns is key to gaining knowledge and insight.

Collectively, our ability to store data now far exceeds the rate at which we are able to understand it. Scientists and domain experts expect to discover patterns, thus to help the further understanding of those mechanical signals in various tasks like classification and anomaly detection. A lot of advances have been made to learn the (long) temporal correlations that are often embedded in time series. Visualization of time series provides a new perspective to explore those complex dynamics more intuitively.

Many visualization methods have been introduced to address time series mining tasks. The line graph is widely used [9], [14]. While the standard time series graph is effective when dealing with a small data space, it is more challenging to perform common tasks on larger data. Pixel plot is explored to represent a time-series as an arrangement of pixels encoded with different hues which encode the underlying data [10], [5]. Besides aggregating the data into temporal segments with lens view, layout based techniques provide a linear mapping by modifying the spatial arrangement of the time while transforming those graphs which enhance the display.

In [6] the authors propose an early importance driven layout scheme while [7] presents a multi-focus zooming approach which maintains context and temporal distance zooming. In contrast to these schemes, [8] compares stack zooming against standard techniques for navigation of temporal data.

However, more complex and noisy real-world time series data is often difficult to observe through visualization because the dynamics are either too complex or unintuitive. To provide more profound insights about the temporal dynamics embedded in time series, one possible solution is to reformulating the data to explicitly extract features for better visual encode of the temporal dependency.

Reformulating the features of time series as visual clues has been studied in physics as a kind of dynamical systems and raised much attention in data mining and machine learning research. One major approach is to build different network structures from time series for visual analytics. Recurrence Networks were proposed to analyze the structural properties of time series from complex systems [4]. They build adjacency matrices from the predefined recurrence functions to interpret the time series as complex networks for visual inspection and metric based classification. [15] extended the recurrence plot paradigm for time series classification using compression distance. Another way to build a weighted adjacency matrix is to model the temporal-spatial transition dynamics with first order Markov matrix [3]. Although these maps demonstrate distinct topological properties among different time series to visualize the intrinsic statistics, it remains unclear how these topological properties relate to the original time series to help visual analytics as they have no exact inverse operations.

In [16], the author proposed a novel encoding framework to transform time series to images as the input for deep convolution neural networks and achieve the state-of-the-art classification performance. We found proposed temporal Markov encoding maps have exact/approximate inverse maps, making such transformation more interpretable with graph visualization techniques.

Inspired by the temporal Markov encoding approaches, we design a novel visualization framework to translate time series into complex networks while preserve both temporal ordering and statistical dynamics. Our encoding map also has a natural inverse, which makes it possible to visually explore the interesting patterns in the generated graph and match them

back to the original data. Our experimental results suggest the effectiveness on various tasks such as system identification, classification and anomaly detection on both benchmark and real time series data.

II. DATA TRANSFORMATION AND QUANTITATIVE ENCODING

Temporal and frequency correlation are major dependencies embedded in time series data. To build a comprehensive but intuitive visualization, the extracted features from the designed data transformation framework should be able to represent the dynamics in both time and frequencies while there exists a reverse operation to map the information back to raw time series. In this section, we will introduce how to encode the dynamical frequency information in the temporal ordering step by step (Figure 1).

A. Quantization and Markov Matrix

This framework is similar to [3] for encoding dynamical transition statistics. We develop that idea by representing the Markov transition probabilities sequentially to preserve information in the temporal dimension. The first step is to quantize time series to build the first order Markov Matrix.

Given a time series $X = \{x_1, x_2, \dots, x_n\}$, we need to quantize its values in Q bins. Quantile is the most common way to achieve a discretized dictionary. By identifying the Q quantile bins, each value x_i is mapped to each q_i . In [13], a different quantization approach is proposed. Based on the empirical observation that normalized time series has a Gaussian distribution [11], it is desirable to have a discretization technique that will produce bins with approximate equiprobability. We can simply determine the breakpoints that will produce equal-sized areas under Gaussian curve. Breakpoints are a sorted list of numbers $Q = q_1, q_2, \dots, q_Q$ such that the area under a $N(0, 1)$ Gaussian curve follows $P(q_{i+1}) - P(q_i) = \frac{1}{Q}$.

Quantile mapping ignores the Gaussian priori, so it is more data-specific while the distribution on different datasets tends to vary significantly, although some data comes from the similar distribution. Gaussian mapping does not have such problems and performs well with Symbolic Aggregate Approximation (SAX) in classification and query tasks. In the next sections, we use Gaussian mapping to create the dictionary for quantizing time series.

After assigning each x_i to its corresponding bin q_j ($j \in [1, Q]$), we construct a $Q \times Q$ weighted adjacency matrix W by counting transitions among bins in the manner of a first-order Markov chain along each time step. $w_{i,j}$ is the frequency of a point in the bin q_j followed by a point in the bin q_i . After normalization by $\sum_j w_{ij} = 1$, W is a Markov matrix:

$$W = \begin{bmatrix} w_{11}|P(x_t \in q_1|x_{t-1} \in q_1) & \cdots & w_{1Q}|P(x_t \in q_1|x_{t-1} \in q_Q) \\ w_{21}|P(x_t \in q_2|x_{t-1} \in q_1) & \cdots & w_{2Q}|P(x_t \in q_2|x_{t-1} \in q_Q) \\ \vdots & \ddots & \vdots \\ w_{Q1}|P(x_t \in q_Q|x_{t-1} \in q_1) & \cdots & w_{QQ}|P(x_t \in q_Q|x_{t-1} \in q_Q) \end{bmatrix} \quad (1)$$

B. Markov Transition Field and Network Graph

Markov matrix, while incorporating the Markov dynamics, discards the conditional relationship between the distribution of X and the temporal dependency on the time steps t_i . Markov Transition Field (MTF) extends Markov matrix by aligning each probability along the temporal order.

$$M = \begin{bmatrix} M_{11} & M_{12} & \cdots & M_{1n} \\ M_{21} & M_{22} & \cdots & M_{2n} \\ \vdots & \ddots & \ddots & \vdots \\ M_{n1} & M_{n2} & \cdots & M_{nn} \end{bmatrix} = \begin{bmatrix} w_{ij}|x_1 \in q_i, x_1 \in q_j & \cdots & w_{ij}|x_1 \in q_i, x_n \in q_j \\ w_{ij}|x_2 \in q_i, x_1 \in q_j & \cdots & w_{ij}|x_2 \in q_i, x_n \in q_j \\ \vdots & \ddots & \vdots \\ w_{ij}|x_n \in q_i, x_1 \in q_j & \cdots & w_{ij}|x_n \in q_i, x_n \in q_j \end{bmatrix} \quad (2)$$

That is, in the Markov matrix M , the quantile bins that contain the data at time steps i and j (temporal axis) are q_i and q_j ($q \in [1, Q]$). M_{ij} denotes the transition probability of $q_i \rightarrow q_j$. In other words, we spread out matrix W , which contains the transition probability on the magnitude axis, into the W by considering temporal positions.

By assigning the probability from the quantile at time step i to the quantile at time step j at each value M_{ij} , the MTF M encodes multi-scale transition probabilities of the time series. $M_{i,j}|i-j|=k$ denotes the transition probability between the points with time interval k . For example, $M_{ij}|j-i=1$ illustrates the transition process along the time axis with a skip step. The main diagonal M_{ii} , which is a special case when $k = 0$ captures the probability from each quantile to itself (the self-transition probability).

The square matrix M can be interpreted as a network graph $G = (V, E)$ directly. Each value M_{ij} represents the edge weights, each row/column corresponds to the vertex index. Note that such settings allow the inverse operation to match nodes back to the time series, as the mapping between node V_i in the graph and the point x_i in the time series is bijective. As in Figure 1, the nodes in red, blue and yellow exactly match different time points respectively.

C. Dimension Reduction

To make the matrix size manageable for more efficient computation and visualization, two common dimension reduction techniques can be performed before or after the MTF encoding:

- **Piecewise Aggregate Approximation (PAA)** reduces the time series from n dimensions to w dimensions, the data is divided into w equal sized frames. The mean value of the data falling within a frame is calculated and a vector of these values becomes the data-reduced representation. The representation can be visualized as an attempt to approximate the original time series with a linear combination of box basis functions. PAA is a special case of moving average model when the skip size equals to the length of the sliding window.

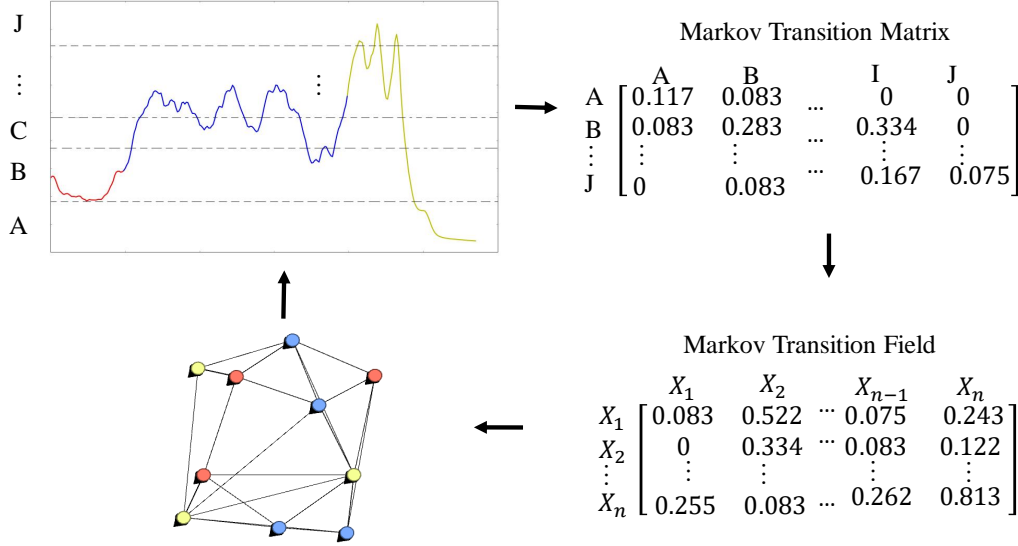


Fig. 1. Illustration of the proposed encoding map of a Markov Transition Field. X is a sequence of the typical time series in the 'ECG' dataset. X is first discretized into Q quantile bins. Then we calculate its Markov Transition Matrix W and finally build its MTF M by Equation. (2). We reduce the image size from 96×96 to 48×48 by averaging the values in each non-overlapping 2×2 patch.

- **Blurring** on MTF is fulfilled by averaging the values in each non-overlapping $m \times m$ patch with a blurring kernel,

typically a Bivariate Gaussian $\{\frac{1}{2\pi\sigma^2}e^{-\frac{x_1^2+x_2^2}{2\sigma^2}}\}_{m \times m}$ or a simple average kernel $\{\frac{1}{m^2}\}_{m \times m}$. That is, we aggregate the transition probabilities in each subsequence of length m together with a convolution operation with a smaller kernel. The final output size $S = \lceil \frac{n}{m} \rceil$.

PAA is simple and effective especially when the detailed structure of the time series can be ignored while the major trends/patterns are important. For analyzing the visual patterns, small information loss and minimum manipulation on the raw data is preferred to preserve the information in the raw data. Hence, we only use Gaussian blurring to reduce the MTF size.

III. VISUAL ENCODING AND INVERSE MAPPING

For general inspection, the graph $G = (V, E)$ generated from MTF has a direct mapping from the vertex V to the time index i in the original time series. It is easy to encode the time index $\{1, 2, \dots, n\}$ as the color of the vertices. Along the temporal axis, we choose PageRank weights as the vertex size and Markov matrix weight as the edge color to observe when (to which vertex) the big information/citation flow occurs.

Modularity is an important pattern in network analysis to identify the specific local structures. To verify the modules and distinct patterns, we also figure out the modularity label and the clustering coefficient [2], [12] for each vertex and map them back to the original time series, and assign the modularity labels to each vertex. In this way, another visual encoding idea is to encode the vertex/edge color as the module labels it belongs to and the vertex size as the clustering coefficient. So we have two visual encoding frameworks for general/flow inspection and modularity discovery (Table I).

TABLE I
VISUAL ENCODING FRAMEWORK

Flow Encoding Color Size	Vertex		Edge
	Time index	PageRank weight	Markov matrix weight Constant
Modularity Encoding Color Size	Vertex		Edge
	Module label	Clustering coefficient	Module label of the target Constant

As each vertex index and the time index are one-to-one mapping, we can easily map the interesting patterns we find in the graph back to the time series. As in Figure 2 (c) and (d), the three communities discovered in both graph have their corresponding shapelets in the raw time series.

IV. EXPERIMENTS AND ANALYSIS

A. System Identification

To highlight the potential of the encoding graph described above, we first apply it to two time series belonging to different dynamical systems (Figure 2 (a)). The first time series is the variable of the chaotic Lorenz equations

$$\begin{aligned} \frac{dx}{dt} &= \sigma(-x + y) \\ \frac{dy}{dt} &= rx - y - xz \\ \frac{dz}{dt} &= bz + xy \end{aligned} \quad (3)$$

with parameter values $\sigma = 10, b = \frac{8}{3}$ and $r = 28$. Numerical solutions of these equations lead to an attractor embedded

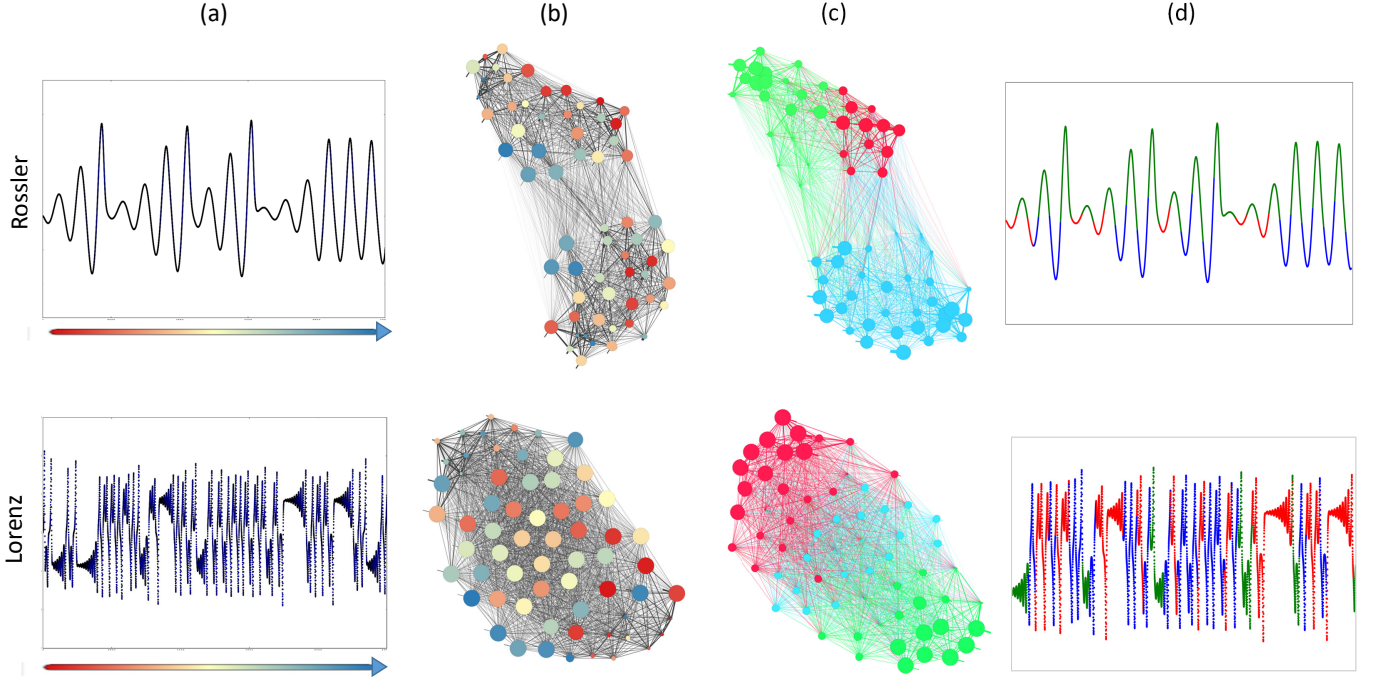


Fig. 2. Network graph generated from Lorenz and Rossler system with $Q = 50$ and $S = 64$. (a). Original time series; (b). Graph with the flow encoding approach; (c). Graph with the modularity encoding approach (community resolution = 1); (d). Time series with the inverse mapping of the graph community.

in a three-dimensional space with coordinates. The trajectory rotates about one of two unstable fixed points then escapes to orbit the other fixed point. This behavior is recognizable in the variable x since its values oscillate between the positive and the negative.

The second time series is the variable of the chaotic Rossler equations

$$\begin{aligned}\frac{dx}{dt} &= -(y+z) \\ \frac{dy}{dt} &= x+ay \\ \frac{dz}{dt} &= b+z(x-c)\end{aligned}\quad (4)$$

with parameter values $a = 0.432$, $b = 2$ and $c = 4$. The trajectory within the attractor follows an outward spiral close to the plane around an unstable fixed point. Once the trajectory spirals out enough, a second fixed point influences it, causing a rise and twist in the z dimension. This behavior generates a

quasi-periodic oscillatory pattern in the x variable, with peaks and troughs in different amplitudes.

From Figure 2, we can clearly identify the difference between these two dynamical systems. The network of Rossler system has a clear sparse connection area like a bipartite graph. The big blue vertices near the boarder of the bipartite gap indicate the homogeneous large transition towards the time end. This means the system has either converging or periodical behavior, which is consistent with the three even peaks at the ends, as the system begins to orbit in a quasi-periodical way. Lorenz system indicates strong coherence without any gap inside the graph. Instead, the big transition flow incurs from the start to the end as we can observe both enough big blue and red vertices. This means the system is very unstable, transiting among states very often without a converging or periodical trend.

The most interesting results are the shapelet discovered from the graph communities, as shown in Figure 3. The com-

community detection algorithm recognizes three modules in both systems. After matching the vertices with different module labels back to the raw time series, the time series shapelets share the common patterns respectively. For Lorenz system, three communities correspond to the high frequency oscillation (red), low frequency oscillation (green) and the alternating behavior switching between those two fixed points (blue). Rossler system is quasi-periodic with steady slow transition among multiple magnitudes. The shapelet mapped from the community can be summarized as three basic 'wavelet' types: the convex wavelets within small (red) and big (blue) magnitude and the concave wavelet (green). Thus, the basic patterns embedded in the time series are discovered through the graph visualization and network statistics.

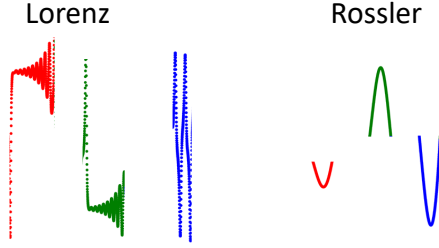


Fig. 3. The basic time series "shapelets" discovered by the community in the graph.

B. Pattern Mining

We test our approach on three typical compound functions [17] with different patterns to test if we can identify those specific modules through the graph and network statistics.

1). The major function is defined by

$$f_1(x) = \begin{cases} 0 & \text{if } x \in [0, 1] \cup [3.5, 4.5] \\ 1 & \text{otherwise} \end{cases}$$

The second pattern in the middle range is generated by the function

$$f_1^*(x) = \begin{cases} 0.25 & \text{if } x \in [2.9, 3] \\ 0.5 & \text{if } x \in [1.6, 1.7] \\ 0.75 & \text{if } x \in [2.2, 2.3] \end{cases}$$

where $x \in X = [0, 4.5]$. x_i is obtained by random sampling 2000 non-repeatable numbers from X with a uniform distribution, and the corresponding output value y_k are computed by $f(x)$ and $f^*(x)$. The overall function is $F_1(x) = f_1(x) + f_1^*(x)$.

2). The major function is defined by

$$f_2(x) = g(x, \frac{1}{6}, \frac{1}{2}, \frac{1}{6})$$

The second pattern with very large derivations is given by

$$f_2^*(x) = g(x, \frac{1}{64}, \frac{1}{4}, \frac{1}{128}) + g(x, \frac{1}{64}, \frac{1}{20}, \frac{1}{128})$$

Where $x \in X = [0, 1]$, g is defined as

$$g(x) = \frac{\alpha}{\sqrt{2\pi}\tau} \cos\left(\frac{(x-\mu)\pi}{\tau}\right) e^{-\frac{(x-\mu)^2}{2\tau^2}} \quad (5)$$

The overall function is $F_2(x) = f_2(x) + f_2^*(x)$. The input values x_i are selected by sampling 2000 numbers from a uniformly distributed grid on X . The corresponding output values y_i are computed by $f(x)$ and $f^*(x)$.

3). Define the first function as

$$F_3(x) = g(x, \frac{1}{5}, \frac{1}{4}, \frac{1}{12}) + g(x, \frac{1}{5}, \frac{3}{4}, \frac{1}{12})$$

The other pattern from the under-sampled points is given by the lower sampling frequency. $x \in X = [0, 1.5]$, the outliers x_i^* are collected by sampling 200 numbers from a uniform distributed grid on $[0, 0.5]$ and 200 numbers from a uniform distributed grid on $[1.0, 1.5]$. The smooth function is generated by 2000 numbers of points from a uniform distributed grid on $(0.5, 1.0)$. So the primary function is $f_3(x) = F_3(x)$, $x \in (0.5, 1)$ and the other pattern is generated by $f_3(x) = F_3(x)$, $x \in [0, 0.5] \cup [1, 1.5]$. The output y_i is computed by $F_3(x)$. The data with 2400 (x_i, y_i) pairs is the function contaminated by the unevenly under-sampled segments.

These functions represent the simple cases of three typical mixed modularities (Figure 4 (a)). The compound patterns in $F_1(x)$ are in the range of the uncontaminated data but appearing at wrong positions. This sometimes occurs when the value in a time series is within normal magnitude but maybe corrupted or inaccurate. $F_2(x)$ simulates the mixed pattern with significant large deviations, indicating the situations where the time series contains some unknown samples with extraordinary bounds. $F_3(x)$ includes seemingly uncontaminated but shifted and under-sampled data. This may happen among heterogeneous observations from several experiments where the time series obtained is highly sparse and biased and the main function needs to get rid of their influence.

The network graphs (Figure 4(b) and (c)) are generated with $Q = \{10, 10, 10\}$ and $S = \{50, 200, 200\}$ respectively. The ground truth of the primary pattern is highlighted in red. To separate the mixed structures in the graph, we select the vertices that are not bundled so closely within its near communities, since the different module has few but distinct patterns in the time series. The rareness and weirdness isolate the corresponding vertices in the graph to be comparatively independent. The recognition process combines two procedures:

- We visually inspect the graph and select the most isolated vertices, denote as the set H .
- The top k vertices with the smallest clustering coefficients are selected, denote as the set S

Then we use the union $H \cup S$ to finally decide which vertices are most isolated (green vertices in Figure 4(c)). The vertices marked as from the different module are mapped back to the original time series to show the corresponding subsequence and their raw patterns. Comparing Figure 4 (a) and (d), our results demonstrate accurate recognition performance on where and what the different patterns are.

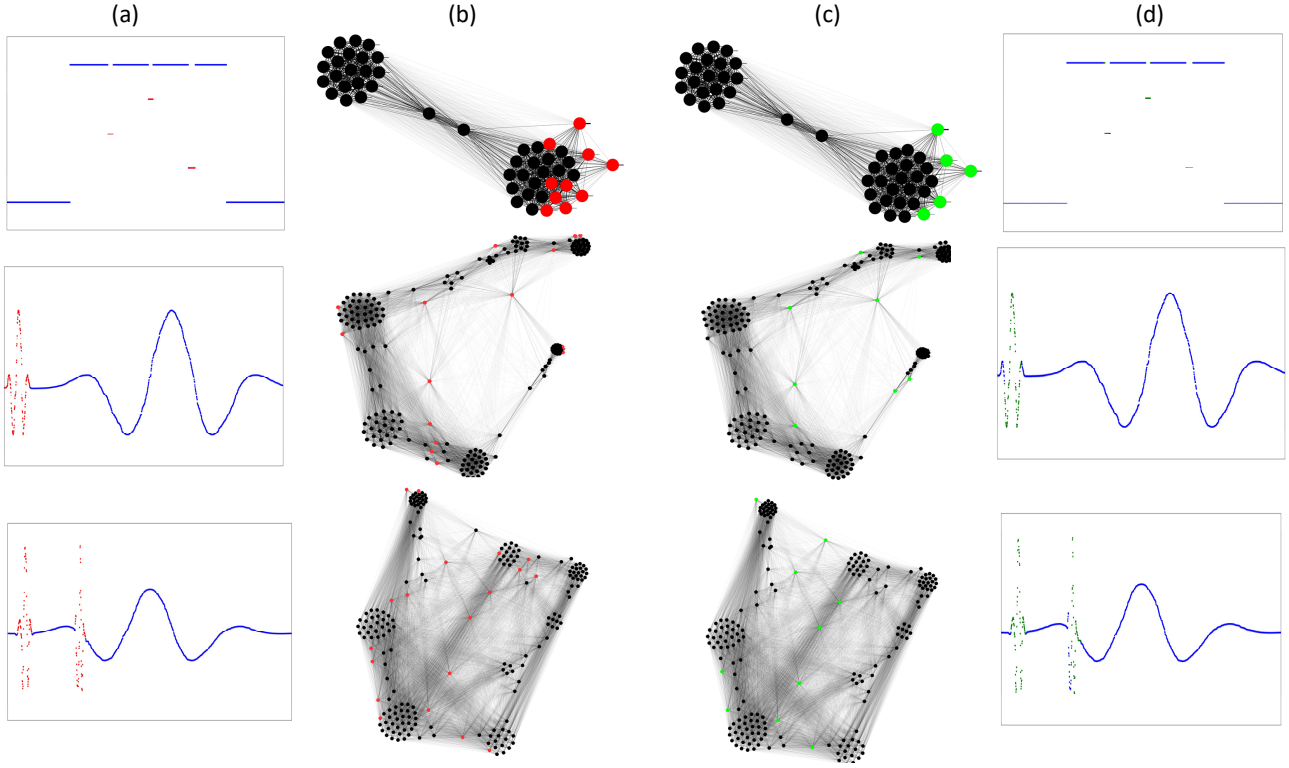


Fig. 4. (a). Signals with the major pattern 1 (blue) and minor pattern 2 (red); (b). Generated graph, ground truth (outlier) is colored by red; (c). Vertex with pattern 1 (green) selected by visual inspection and the smallest clustering coefficients; (d). Inverse mapping from the graph to the original time series, the rare pattern in the time series subsequence obtained from the suspected vertex is colored by green.

Note that sometimes the vertices selected by visual inspection and lowest clustering coefficient are significantly different. Choosing the set intersection $H \cap S$ will sometimes leads to the high false-negative rate as some important but rare patterns might be ignored. This would be the worst case in, for example, anomaly detection. So we choose the union $H \cup S$ to guarantee the low false-negative rate, as the false-positive patterns can be easily eliminated by inversely mapping the vertices to the original time series and comparing the subsequence distances or even by visually inspecting on those mapped shapelets.

C. Classification

Our final experiment is to identify the class labels on three real time series datasets: ECG, Wafer and Gun Point [18]. The ECG dataset is a time series of the measurements recorded by one electrode during one heartbeat. The data has been annotated by cardiologists and a label of normal or abnormal

is assigned to each data record. The Gun Point dataset contains two-dimensional time series extracted from video of two actors either aiming a gun or simply pointing at a target. The two dimensions correspond to the X and Y coordinates of the actors right hand, we only discuss the Y-Axes as X-Axes is not available. The Wafer dataset is recorded by one vacuum-chamber sensor during the etch process of silicon wafers for semiconductor fabrication with the normal or abnormal.

We randomly pick one sample of each label and draw the network graph with $Q = \{50, 50, 10\}$ for the Gun Point, ECG and Wafer dataset. Dimension reduction is skipped as we wish to keep full information in the original data. The network graph uses modularity encoding to highlight the distinct structures for different labels (Figure 5).

An interesting finding is that, even though we cannot observe small differences in the mechanical signals, network graphs can sometimes show us the clear difference in their network structures. For ECG dataset, the label difference is

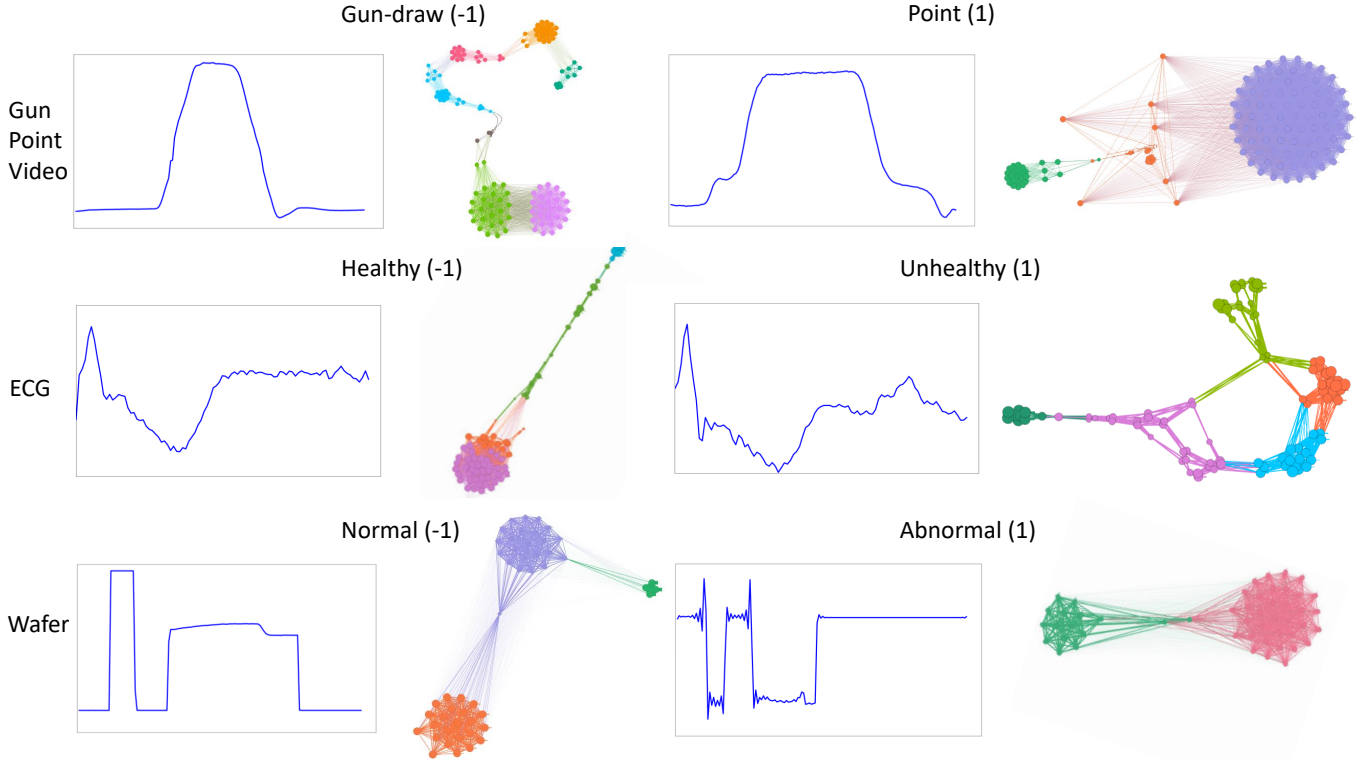


Fig. 5. Original time series and the network graph with the modularity encoding.

TABLE II
NET WORK STATISTICS. ON THE ECG, WAFER AND GUN POINT DATASET, THE MEAN AND STANDARD DEVIATION ACROSS ALL SAMPLES ARE REPORTED. THE BOLD NUMBER MEANS THE DIFFERENCE OF THIS MEASUREMENT BETWEEN LABELS IS STATISTICALLY SIGNIFICANT IN A PAIRED T-TEST

	Avg Degree	AVG Weighted Degree	Diameter	Density	Modularity	# Communities	AVG Clustering Coefficient	AVG Path length
Lorenz	62.06	1.00	2.00	0.99	0.18	3	0.96	1.05
Rosler	48.97	1.01	2.00	0.78	0.41	3	0.87	1.25
3-Notch	51.00	1.00	1.00	1.04	0.45	2	0.99	1.00
FineFeature	130.22	1.01	2.00	0.65	0.71	7	0.83	1.36
UnderSample	79.57	1.00	3.00	0.40	0.72	6	0.88	1.64
ECG (-1)	19.92	6.04	11.40	0.21	0.50	3.1	0.59	4.33
ECG (1)	13.42	4.97	10.00	0.14	0.58	5.2	0.43	3.93
Wafer (-1)	41.36	1.03	2.00	0.84	0.47	2.2	0.94	1.20
Wafer (1)	38.28	1.05	1.80	0.78	0.45	2.5	0.94	1.26
Gun Point (-1)	33.85	19.79	19.80	0.23	0.64	6.8	0.80	8.48
Gun Point (1)	56.07	36.16	13.20	0.38	0.44	2.9	0.85	6.22

more differentiable from the network structures, where the healthy ECG has a linear structure with four close communities, while the unhealthy ECG has a round structure with five vague communities.

For the Gun Point dataset, the time series seems to be very similar as the Point pattern is stretched from the Gun-draw to have a wider plateau in the middle. However, their network structures show a huge difference with seven and three clear communities, respectively.

D. Network Statistics and Summary

Our encoding network graph not only supports the purpose of visualization of time series, it also bridges the visual

analytics, network dynamics and time series analysis with a natural inverse map from network graph to the original time series. Networks can be analyzed by exploring an extensive set of statistical properties of the associated time series with an active visual understanding. For example, motifs discovered in a network are mapped as specific spatial-temporal patterns in a time series, which are characterized by looking at the corresponding forward and backward maps. At the same time, different dynamical regimes in time series can be visualized and understood by exploring an extensive set of topological statistics at the associated network domain.

Table II summarizes some common network statistics on

all of our experimental datasets [1]. On the ECG, Wafer and Gun Point dataset, the average across all the samples with the specific labels is reported.

Specifically, some statistics associated with different labels are significantly different. The average degree and weighted degree are the right features to distinguish between the negative and positive labels across all three datasets. This means that the network graph generated from each label has different overall network topology. The network density and the number of communities are also helpful in classifying healthy and unhealthy samples for the ECG dataset. For the Gun Point dataset, all statistics show significant difference between the action of Gun-draw and Point. Our network graph captures the essential features of the spatial-temporal dynamics to describe those two actions. Rather than user study, the network statistics also provide a more objective and efficient way, together with the inspection of the visualization graphs, to verify the effectiveness of our visualization approach across all the three tasks.

V. CONCLUSION AND FUTURE WORK

In this work, we present a novel approach to translate time series into the network graph for visualization and pattern mining. The proposed graph is able to encode both temporal ordering and statistical dynamics in the original time series through the Markov Transition Field (MTF). Our encoding map also has a natural inverse, enable the visual exploring of the interesting patterns in the generated graph and the inverse mapping to the original data. The experiments on both synthetic and real time series data demonstrate the effectiveness of our visualization and mining framework on the tasks of system identification, classification and pattern mining. The inverse mapping is one-to-one, which provides a link between pattern recognition in the network graph and time series analysis. Moreover, our work bridges the gap between visual analytics, time series data mining and graph statistics, as the generated network can be analyzed by exploring an extensive set of statistical properties of the associated time series with an active visual understanding.

As one of the future work, we are interested in designing a unified user interface to build a set of interaction mechanism to enable the active exploring of the time series through the network graph by the users. The bin size could also be determined by minimizing the reconstruction loss through sampling from the MTF. Another interesting work to do is to deeply incorporate the statistical complex network theory with the visual analytics framework to provide theoretical support and further reinforce the profile of time series visualization.

REFERENCES

- [1] M. Bastian, S. Heymann, M. Jacomy, et al. Gephi: an open source software for exploring and manipulating networks. *ICWSM*, 8:361–362, 2009.
- [2] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.
- [3] A. S. Campanharo, M. I. Sire, R. D. Malmgren, F. M. Ramos, and L. A. N. Amaral. Duality between time series and networks. *PloS one*, 6(8):e23378, 2011.
- [4] R. V. Donner, Y. Zou, J. F. Donges, N. Marwan, and J. Kurths. Recurrence networks: a novel paradigm for nonlinear time series analysis. *New Journal of Physics*, 12(3):033025, 2010.
- [5] M. Hao, U. Dayal, D. Keim, and T. Schreck. Multi-resolution techniques for visual exploration of large time-series data. In *Proceedings of the 9th Joint Eurographics / IEEE VGTC Conference on Visualization, EUROVIS'07*, pages 27–34, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.
- [6] M. C. Hao, U. Dayal, D. A. Keim, and T. Schreck. Importance-driven visualization layouts for large time series data. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.*, pages 203–210. IEEE, 2005.
- [7] W. Javed and N. Elmqvist. Stack zooming for multi-focus interaction in time-series data visualization. In *2010 IEEE Pacific Visualization Symposium (PacificVis)*, pages 33–40. IEEE, 2010.
- [8] W. Javed and N. Elmqvist. Stack zooming for multifocus interaction in skewed-aspect visual spaces. *IEEE transactions on visualization and computer graphics*, 19(8):1362–1374, 2013.
- [9] W. Javed, B. McDonnell, and N. Elmqvist. Graphical perception of multiple time series. *IEEE transactions on visualization and computer graphics*, 16(6):927–934, 2010.
- [10] R. Kincaid and H. Lam. Line graph explorer: Scalable display of line graphs using focus+context. In *Proceedings of the Working Conference on Advanced Visual Interfaces, AVI '06*, pages 404–411, New York, NY, USA, 2006. ACM.
- [11] R. J. Larsen and M. L. Marx. *An introduction to mathematical statistics and its applications*. 1986.
- [12] M. Latapy. Main-memory triangle computations for very large (sparse (power-law)) graphs. *Theoretical Computer Science*, 407(1):458–473, 2008.
- [13] J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 2–11. ACM, 2003.
- [14] P. McLachlan, T. Munzner, E. Koutsofios, and S. North. Liverac: interactive visual exploration of system management time-series data. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1483–1492. ACM, 2008.
- [15] D. F. Silva, V. Souza, M. De, and G. E. Batista. Time series classification using compression distance of recurrence plots. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, pages 687–696. IEEE, 2013.
- [16] Z. Wang and T. Oates. Imaging time-series to improve classification and imputation. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.
- [17] Z. Wang, T. Oates, and J. Lo. Adopting robustness and optimality in fitting and learning. *arXiv preprint arXiv:1510.03826*, 2015.
- [18] L. Wei and E. Keogh. Semi-supervised time series classification. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 748–753. ACM, 2006.